Capítulo V

DETERMINACIÓN DE REQUERIMIENTOS

Cuando un cliente solicita que se contruya un nuevo sistema, tiene alguna noción de lo que el sistema debe hacer. Cada sistema basado en software tiene un propósito, usualmente expresado como algo que el sistema debe hacer. Un *requerimiento* es una característica del sistema o una descripción de algo que el sistema es capaz de hacer con el objeto de satisfacer el propósito del sistema.

La Figura 5.1 explica el proceso de determinación de los requerimientos para un sistema basado en software. En primer lugar se trabaja con los clientes, para extraer los requerimientos, formulando preguntas, haciendo demostraciones de sistemas similares y hasta desarrollando prototipos de todo o partes del sistema propuesto. Después se capturan dichos requerimientos en un documento o en una base de datos. En primer término se escriben los requerimientos, de modo que clientes y desarrolladores puedan ponerse de acuerdo acerca de lo que el sistema debe hacer. Por lo general, los requerimientos se escriben nuevamente, en una representación más matemática, para que los diseñadores puedan transformar los requerimientos en un buen diseño del sistema. Un paso de verificación asegura que los requerimientos sean completos, exactos y consistentes, y un paso de validación garantiza que lo descrito es lo que el cliente pretende ver en el producto final.

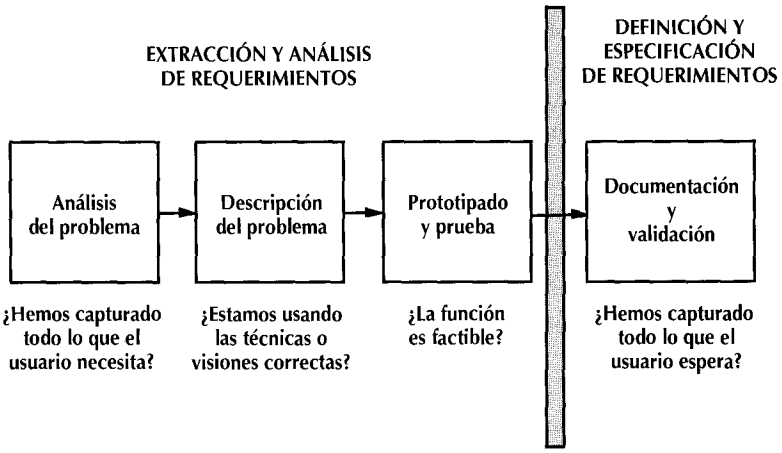
EXTRACCIÓN Y ANÁLISIS DE REQUERIMIENTOS
DEFINICIÓN Y ESPECIFICACIÓN DE REQUERIMIENTOS
Análisis del problema
Descripción del problema
Prototipado y prueba
Documentación y validación
¿Hemos capturado todo lo que el usuario necesita?
¿Estamos usando las técnicas o visiones correctas?
¿La función es factible?
¿Hemos capturado todo lo que el usuario espera?

Figura 5.1 El proceso de determinación de los requerimientos.

La falta de cuidado en la comprensión, la documentación y la gestión de los requerimientos puede llevar a una miríada de problemas: construir un sistema que resuelve el problema equivocado, que no funciona como se espera o que presenta dificultades para que los usuarios puedan comprenderlo y utilizarlo. Es más, un proceso de requerimientos mediocre puede en realidad resultar muy caro. Así que es rentable tomar el tiempo que sea necesario para comprender el problema y su contexto y obtener los requerimientos correctos desde el primer momento.

La extracción de los requerimientos es una parte especialmente crítica del proceso. Podemos utilizar una variedad de técnicas para determinar qué es lo que los usuarios y los clientes quieren realmente. A veces el trabajo consiste en la automatización de un sistema manual, de modo que es fácil examinar lo que ya está hecho. Pero a menudo debemos trabajar con los usuarios y los clientes, para comprender un problema cuando todavía no se ha encontrado una solución. Debemos analizar el problema antes de considerar cualquier solución, lo que por lo general se hace desglosando el problema en piezas pequeñas más fáciles de comprender. Una de las maneras de realizar un análisis de problema consiste en identificar las personas, los procesos y los recursos involucrados, y después documentar las relaciones que existen entre ellos. e les pregunta a los clientes quién está involucrado y se intenta determinar el límite del sistema. e averigua qué elementos de datos pasan de un rol a otro, y cuáles procesos transforman los datos de una forma o estado a otro. Durante la extracción de los requerimientos se interroga a usuarios y clientes sobre los mismos aspectos, de muchas maneras, hasta tener la seguridad de que se comprende lo que quieren y necesitan.

Por lo general resulta til separar los requerimientos en tres categorías:

1. equerimientos que deben ser absolutamente satisfechos.
. equerimientos que son muy deseables pero no indispensables.
3. equerimientos que son posibles, pero que podrían eliminarse.

El análisis de requerimientos por categoría es til para que todos los participantes comprendan lo que realmente se necesita. ambién es til cuando el proyecto está restringido en tiempo o recursos por lo tanto, si el sistema tal como se encuentra definido costará demasiado o tomará demasiado tiempo desarrollarlo, los requerimientos de categoría 3 se dejan de lado y los de categoría pueden analizarse para su eliminación o postergación.

Cada uno de los requerimientos de un sistema trata acerca de los *o eto* o *enti e* , los *e t o* en que estos pueden estar y las *un ione* que se realizan para cambiar los estados o las características del objeto.

odos los requerimientos son descripciones específicas de funciones o características que conducen al propósito más general del sistema. Por lo tanto, se buscan los requerimientos que definen los objetos del sistema un empleado es una persona que es pagada por la compañía , que los limitan un empleado será pagado por no más de horas de trabajo por semana , o define relaciones entre ellos un empleado está supervisado por el empleado si puede autorizar el cambio del salario de x .

ótese que ninguno de estos requerimientos especifica cómo debe implementarse el siste ma. En otras palabras, no se hace mención de qué sistema de gestión de base de datos utilizar, de cuánta memoria debe tener la computadora o qué lenguaje de programación se debe aplicar para desarrollar el sistema. Estas descripciones específicas de la implementación no se consideran como requerimientos, a menos que así lo establezca el cliente. Es decir, un requerimiento apunta al propósito del sistema sin considerar cómo se va a implementar. Cuando los requerimientos se contemplan con esta luz, es fácil ver que algunas características del sistema pueden ser irrele vantes. En particular deben eliminarse de cualquier especificación de requerimientos aquellas características que no tienen nada que hacer con el propósito del sistema. e dice que los requeri mientos identifican el qué del sistema, en tanto el diseño establece el cómo del sistema.

Esta distinción se vuelve todavía más clara si tenemos en mente el propósito de la extracción y análisis de los requerimientos. La Figura 5. muestra dónde se acomodan estas actividades en el amplio contexto del desarrollo del sistema. Los requerimientos se extraen al comienzo del desarrollo siendo nuestra meta la determinación de la naturaleza del problema del cliente. Discutir sobre cualquier solución es prematuro hasta tanto el problema no esté definido

con claridad. Es más, el problema se expresa más fácilmente en términos del negocio del cliente. De este modo, los requerimientos deben centrarse en el cliente v el problema, no sobre la implementación o la solución.

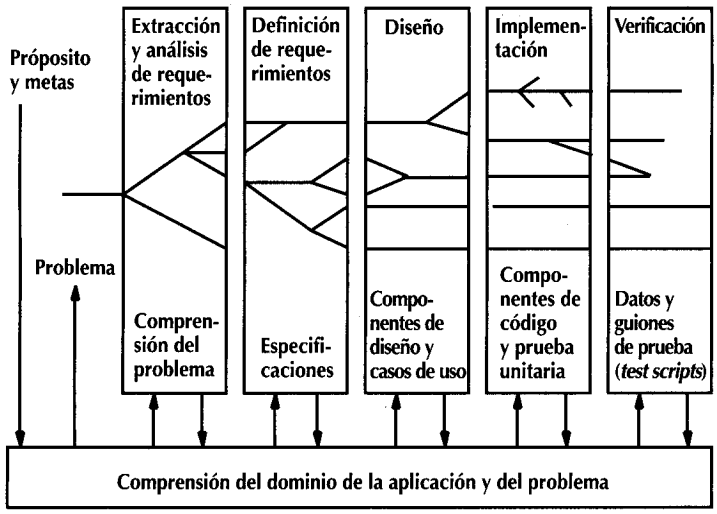

Figura 4.2 Actividades del proceso de desarrollo del software

ay que procurar que los requerimientos sean verificables. De esta forma, una vez que un requerimiento ha sido declarado, se pueden examinar todas las posibles entidades y las actividades que podrían satisfacer el requerimiento, dividiéndolas en dos clases: aquellas que satisfacen el requerimiento y las que no. Esta partición debe ser repetible, los miembros de la clases no deben variar de acuerdo con quien está haciendo la clasificación.

La capacidad de verificación facilidad de medición debe tratarse apenas se extraen los requerimientos. ntentaremos cuantificar las maneras en que se puede probar si una solución potencial satisface un requerimiento dado. Estos criterios de encaje forman una descripción objetiva del significado del requerimiento cuando tales criterios no pueden ser fácilmente expresados es probable que los requerimientos sean ambiguos, incompletos o incorrectos.

Por ejemplo, un cliente puede enunciar un requerimiento de esta manera:

La información sobre calidad del agua debe estar accesible en forma inmediata.

El cliente probablemente tenga una clara idea acerca de lo que significa inmediatamente y es ese concepto el que debe ser capturado en el requerimiento. Podemos expresarlo con mayor exactitud diciendo:

Los registros de calidad del agua deben ser recuperados dentro de los cinco segundos de la solicitud.

Es esta segunda formulación del requerimiento la que puede ser probada objetivamente: se efect a una serie de solicitudes y los registros deben ser suministrados por el sistema dentro de los cinco segundos de cada solicitud.

Existen tres maneras que ayudan a hacer requerimientos verificables:

Especificar una descripción cuantitativa para cada adverbio o adjetivo de modo que el significado de cada calificador sea claro y no ambiguo.

eemplazar los pronombres por los nombres específicos de las entidades.

Asegurar que todo sustantivo esté definido exactamente en un lugar en los documentos de los requerimientos.

es de documentos de requerimientos

Debido a que el foco de la atención está puesto sobre el problema del cliente, la extracción y el análisis de requerimientos sirve a dos propósitos diferentes pero relacionados. Por una parte, la extracción nos permite escribir un documento de definición de requerimientos (Documento de Requerimientos de Usuario o DRU); escrita en términos que el cliente puede entender, la definición de requerimientos es un listado completo de todas las cosas que el cliente espera que haga el sistema propuesto. Representa una comprensión entre el cliente y el desarrollador de lo que el cliente necesita o desea, y por lo general es escrito en forma conjunta por el cliente y el desarrollador. Por la otra parte, la especificación de requerimientos (Documento de Requerimientos de Software o DRS) reitera la definición en los términos técnicos apropiados para el desarrollo del diseño de un sistema; es la contrapartida técnica al documento de definición de requerimientos y es escrito por analistas de requerimientos. A veces un único documento puede servir para ambos propósitos, lo que lleva a un entendimiento común entre clientes, analistas de requerimientos y diseñadores. Pero a menudo se necesitan ambos documentos, y debemos ser especialmente cuidadosos para que no se produzcan pérdidas de información ni se introduzcan cambios cuando se reinterpreta la definición como una especificación.

.Existe una correspondencia directa entre cada requerimiento del documento de definición y aquellos documentados en la especificación. Es aquí donde comienza la aplicación de los métodos de gestión de configuración usados a lo largo del ciclo de vida. La gestión de configuración es un conjunto de procedimientos que rastrean:

- Los requerimientos que definen lo que el sistema debe hacer
- Los módulos de diseño que se generan a partir de los requerimientos
- El código del programa que implementa el diseño
- Las pruebas que verifican la funcionalidad del sistema
- Los documentos que describen el sistema

En un sentido, la gestión de configuración proporciona las cuerdas que ligan las partes del sistema entre sí, unificando los componentes que se han desarrollado por separado. Estas cuerdas son las que permiten coordinar las tareas de desarrollo, como muestran las trazas horizontales entre las entidades en la Figura 5.2. En particular, durante la extracción y el análisis de los requerimientos, la gestión de configuración detalla la correspondencia entre los elementos de la definición de requerimientos y aquellos de la especificación de requerimientos, de modo que la visión del cliente esté unida a la visión del desarrollador en una forma organizada y rastreable. Si no se definen estas líneas, no hay manera de diseñar casos de prueba para determinar si el código satisface los requerimientos.

5.1.2 Requerimientos funcionales y no funcionales

Los requerimientos describen el comportamiento de un sistema. A medida que el sistema actúa sobre los datos o las instrucciones, los objetos y las entidades se mueven desde un estado de ser a otro: por ejemplo, de lleno a vacío, de ocupado a libre, o de enviando a recibiendo. Esto es, en un estado dado, el sistema satisface un conjunto de condiciones; cuando el sistema actúa puede cambiar su estado global cambiando el estado de un objeto. Los requerimientos expresan los estados del sistema y de los objetos y las transiciones de un estado a otro. En particular, los requerimientos describen las actividades del sistema como una reacción a la entrada y el estado de cada entidad en el sistema antes y después de que ocurra dicha actividad. Por ejemplo, en un sistema de nómina de pago el empleado puede existir en al menos dos estados: empleados aún no pagados y empleados pagados. Los requerimientos describen cómo al emitir los cheques de pago un empleado pasa del primero al segundo estado.

Los requerimientos pueden pensarse de dos maneras: *funcional* y *no funcional*, lo que ayuda a describirlos. Un *requerimiento funcional* describe una interacción entre el sistema y su ambiente. Por ejemplo, para determinar los requerimientos funcionales se decide cuáles estados son aceptables para el sistema. Además, los requerimientos funcionales describen cómo debe comportarse el sistema ante determinado estímulo. Por ejemplo, para un sistema que imprime cheques semanales de pago, los requerimientos funcionales deben responder a las preguntas acerca de cuándo se emiten los cheques. ¿Qué entrada es necesaria para que un cheque se imprima? ¿Bajo qué condiciones puede cambiarse el monto del pago? ¿Qué provoca la remoción de un empleado de la nómina de pagos? Se pueden utilizar diversas técnicas para determinar los requerimientos funcionales, incluyendo los casos de uso.

Una de las maneras más convenientes para la determinación de los requerimientos funcionales de un sistema es la identificación de sus casos de uso. Los casos de uso particionan al sistema en un conjunto de piezas lógicas, mínimamente relacionadas, cada una de las cuales describe alguna de las maneras en que funcionará el sistema. Hay varias ventajas de la visión de un sistema en términos de sus casos de uso:

- Debido a que existen pocas conexiones entre casos de uso, puede examinarse cada caso de uso en forma separada, y comprenderlo sin tener que conocer todos los detalles del sistema más grande. En particular, un tipo de usuario puede comprender la funcionalidad propuesta sin que tenga que aprender sobre la funcionalidad de los otros usuarios.
- Se pueden aplicar los casos de uso como base para estimar cuánto tiempo y esfuerzo se necesitará para diseñar y codificar el sistema.
- El desarrollo del sistema puede ser rastreado en términos de los casos de uso. Esto es, los gerentes pueden seguir el avance de cada caso de uso a medida que es diseñado, codificado y probado.

Las preguntas destinadas a determinar los requerimientos funcionales tienen respuestas que son independientes de la implementación de una solución para el problema del cliente. Se describe lo que el sistema hará sin considerar la computadora particular que se podría utilizar, ni el lenguaje de programación empleado, ni las estructuras internas de datos involucradas, o la clase de papel en que se imprimirán los cheques. Es evidente que en lugar de decirnos qué hará el sistema, este tipo de requerimientos impone restricciones al sistema. Esto es, un *requerimiento no funcional* o *restricción* describe una restricción sobre el sistema que limita nuestras elecciones en la construcción de una solución al problema. Por ejemplo, pueden decirnos que el sistema debe ser desarrollado sobre una computadora Compaq, o que los cheques del pago deben entregarse a los empleados no más de cuatro horas después de la lectura inicial de los datos. De manera similar, pueden decirnos que las consultas al sistema deben ser respondidas dentro de los tres segundos posteriores. Estas restricciones por lo general estrechan nuestra selección de lenguaje, plataforma o técnicas y herramientas de implementación; sin embargo, la selección se realiza en la etapa de diseño, después de haber especificado los requerimientos.

Tanto los requerimientos funcionales como los no funcionales son obtenidos del cliente de manera formal y cuidadosa. Esta extracción formal de los requerimientos es necesaria porque los clientes no siempre son buenos para describir con exactitud lo que desean o necesitan, y los desarrolladores no siempre son buenos para comprender los intereses del negocio de los demás. Los clientes conocen su negocio pero no siempre pueden describir sus problemas de negocios a terceros; las descripciones están llenas de expresiones en jerga y de suposiciones con las cuales los desarrolladores no están familiarizados. Por otra parte, los desarrolladores conocen las soluciones de computación, pero no siempre saben cómo afectarán las soluciones posibles a las actividades de negocios de los clientes. También tienen su propia jerga y suposiciones, y a menudo creen estar hablando el mismo lenguaje, cuando de hecho dan significados diferentes a la misma cosa. Así, si la comunicación entre desarrolladores y clientes no está cuidadosamente

organizada y fomentada, puede suceder que una especificación sea malentendida o quede incompleta.

5.2 ASPECTOS A CONSIDERAR EN LA DEFINICIÓN Y ESPECIFICACIÓN DE REQUERIMIENTOS

Los documentos de definición y especificación de requerimientos describen cómo el sistema interactúa con su ambiente, incluyendo los siguientes aspectos:

Ambiente físico

- ¿Dónde está el equipamiento que necesita el sistema para funcionar?
- ¿Existe una localización o varias?
- ¿Existen restricciones ambientales, tales como temperatura, humedad o interferencia magnética?

Interfaces

- ¿La entrada proviene de uno o más sistemas? • ¿La salida va a uno o más sistemas?
- ¿Existe una manera prescrita en que deban formatearse los datos? • ¿Existe un medio prescrito que los datos deban utilizar?

Usuarios y factores humanos

- ¿Quién usará el sistema?
- ¿Habrá varios tipos de usuario?
- ¿Cuál es el nivel de habilidad de cada tipo de usuario?
- ¿Qué clase de entrenamiento requerirá cada tipo de usuario?
- ¿Cuán fácil le será a un usuario comprender y utilizar el sistema?
- ¿Cuán difícil le resultará a un usuario hacer un uso indebido del sistema?

Funcionalidad

- ¿Qué hará el sistema?
- ¿Cuándo lo hará?
- ¿Existen varios modos de operación?
- ¿Cómo y cuándo puede cambiarse o mejorarse un sistema?
- ¿Existen restricciones de la velocidad de ejecución, tiempo de respuesta o rendimiento?

Documentación

- ¿Cuánta documentación se requiere?
- ¿Debe estar en línea, en papel o en ambos?
- ¿A qué audiencia está orientado cada tipo de información?

Datos

- ¿Cuál será el formato de los datos tanto para la entrada como para la salida?
- ¿Cuán a menudo serán recibidos o enviados?

- ¿Cuán exactos deben ser?
- ¿Con qué grado de precisión deben hacerse los cálculos?
- ¿Cuántos datos fluyen a través del sistema?
- ¿Debe retenerse algún dato por algún período de tiempo?

Recursos

- ¿Qué recursos materiales, personales o de otro tipo se requieren para construir, utilizar y mantener el sistema?
- ¿Qué habilidades deben tener los desarrolladores?
- ¿Cuánto espacio físico será ocupado por el sistema?
- ¿Cuáles son los requerimientos de energía, calefacción o acondicionamiento de aire?
- ¿Existe un cronograma prescrito para el desarrollo?
- ¿Existe un límite sobre la cantidad de dinero a gastar en el desarrollo o en hardware y software?

Seguridad

- ¿Debe controlarse el acceso al sistema o a la información?
- ¿Cómo se podrán aislar los datos de un usuario de los de otros?
- ¿Cómo podrán aislarse los programas de usuario de los otros programas y del sistema operativo?
- ¿Con qué frecuencia deben hacerse las copias de respaldo *(backup)?* • ¿Las copias de respaldo deben almacenarse en un lugar diferente?
- ¿Deben tomarse precauciones contra el fuego, el daño provocado por agua o el robo?

Aseguramiento de la calidad

- ¿Cuáles son los requerimientos para la confiabilidad, disponibilidad, facilidad de mantenimiento, seguridad y los restantes atributos de calidad.
- ¿Cómo deben demostrarse las características del sistema a terceros?
- ¿Debe el sistema detectar y aislar defectos?
- ¿Cuál es el promedio de tiempo prescrito entre fallas?
- ¿Existe un tiempo máximo permitido para la recuperación del sistema después de una falla?
- ¿Cómo puede el sistema incorporar los cambios al diseño?
- ¿El mantenimiento corregirá meramente los errores, o incluirá también el mejoramiento del sistema?
- ¿Qué medidas de eficiencia se aplicarán al uso de recursos y al tiempo de respuesta?
- ¿Cuán fácil debe ser mover el sistema de una ubicación a otra o de un tipo de computadora a otro?

La Figura 5.3 muestra como los requerimientos pueden extraerse de muchas maneras. Podemos ampliar este concepto siendo creativos en la forma de averiguar qué es lo que quieren los clientes. Por ejemplo, se puede:

- revisar la situación actual
- trabajar en el ámbito del usuario para comprender el contexto, los problemas y las relaciones

- entrevistar a los usuarios actuales y potenciales
- realizar un video para mostrar cómo podría funcionar el nuevo sistema • investigar los documentos existentes
- realizar lluvias de ideas con los usuarios actuales y potenciales
- observar las estructuras y los patrones

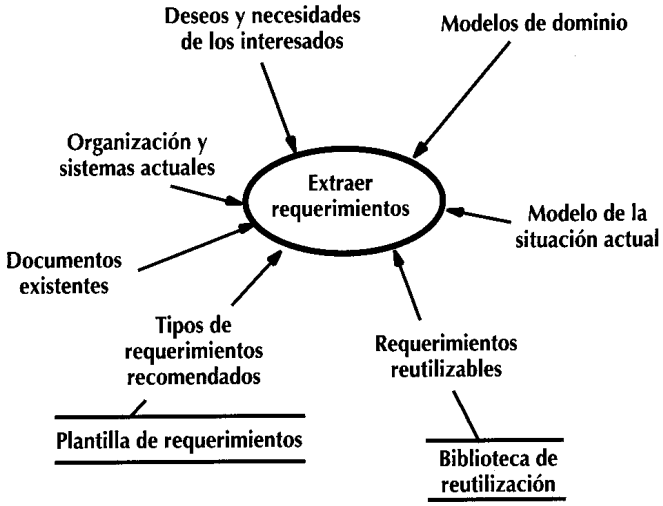

Figura 5.3 Fuentes de posibles requerimientos

5.3 EL ANÁLISIS DE REQUERIMIENTOS Y LOS ESTÁNDARES DE CALIDAD DE SOFTWARE

En SW-CMM, se habla de requerimientos del sistema para referirse a la especificación completa del software a desarrollar. No obstante, es frecuente observar en estándares internacionales de desarrollo de software que dividen el proceso de análisis en dos: la primera fase corresponde a la fase de requerimientos de usuario, y la segunda, a requerimientos de software.

En la fase de requerimientos de usuario se especifican los requerimientos que el sistema de software debe cumplir, utilizando el lenguaje del usuario. Los analistas deben tener especial preocupación por entender el problema en función del lenguaje de sus usuarios, con el propósito de apoyarlos. El conjunto de requerimientos especificados de esta forma son presentados en un documento, llamado "Documento de Requerimientos de Usuario", o DRU.

Por otro lado, en la fase de requerimientos de software, los analistas deben transformar los requerimientos de usuario en un conjunto de requerimientos que especifiquen las características del software, utilizando lenguaje técnico. Dicha labor termina con la aprobación del "Documento de Requerimientos de Software", también conocido por su sigla DRS.

El proceso de captura de los requerimientos se realiza a través de entrevistas y cuestionarios al usuario, con el propósito de determinar de forma más precisa los requerimientos del sistema. Esta actividad debe ser iterativa, ya que los requerimientos irán cambiando a medida que los usuarios y analistas van estructurando y entendiendo mejor los conceptos en discusión.

Como salidas de este proceso se encuentra todo tipo de información que permita apoyar el proceso de análisis y especificación de los requerimientos de usuario. Específicamente, se tienen documentos, cintas de audio o de video con reuniones, tablas, gráficos, anotaciones, etc.

5.3.1 Es ecificación de requerimientos de usuario

Dentro de los requerimientos de usuario, es posible distinguir dos tipos diferentes:

- *equerimientos uncionales*. Son declaraciones de los servicios que proveerá el sistema y de cómo se comportará ante determinado estímulo.
- *equerimientos o uncionales*. Son restricciones de los servicios o funciones ofrecidos por el sistema. ncluyen restricciones de tiempo, sobre el proceso de desarrollo, estándares, etc.

Cada uno de los requerimientos de usuario puede tener asociado uno o más atributos que complementan su información, y que ayudan a entender el requisito, y luego a monitorear su adecuado tratamiento durante todo el proceso de desarrollo. Algunos de los atributos considerados en el estándar ESA PSS-05-0 son:

- *entifica or*: De la forma "RU x", donde x es un número único asociado a cada requisito de Usuario.
- *escripción el requisito*: Consiste en una explicación en español del requisito, explicando en forma no ambigua, qué restricciones impone el requisito sobre el sistema.
- *ecesi a* : Qué tan importante es el requisito en el proyecto. Si es muy importante, entonces se estipula como un requisito esencial, de lo contrario es no-esencial.
- *riori a* : Puede ser alta, media o baja. En cierta medida, diferencia entre los requerimientos esenciales (prioridad alta), los requerimientos normales (prioridad media) y los requerimientos de menor importancia (prioridad baja).
- *stabili a* : Algunos requerimientos se mantendrán estables a lo largo del proyecto, otros estarán sujetos a cambios. Estos cambios podrán darse en las fases DA y DD. Por esto, es necesario hacer una revisión sobre todos los requerimientos a lo largo del proyecto.
- *uente e información*: Aquí se debe citar de dónde proviene el requisito (requisito de usuario, persona, documento, grupo, entrevista, etc.), haciendo referencia a dicha fuente.

Adicionalmente, los requerimientos deben tener ciertos atributos implícitos, los cuales también deben ser tomados en cuenta y descritos dentro del requisito:

- *lari a* : El requisito debe ser claro y entendible por toda persona que lea la especificación. Por lo mismo, debe existir sólo una interpretación válida para el requisito especificado.
- *erificabili a* : Debe ser posible de checar que el requisito esté presente, de asegurarse de que el software tendrá implementando el requisito y que se puede probar el requisito dentro del software implementado.

5.3.2 E em los de requerimientos de usuarios

A continuación se mostrarán tres ejemplos de requerimientos de usuario, el primeros y segundo son funcionales y el tercero es no funcional:

: El Sistema debe ser capaz de generar el número de matrícula de manera automática o dar la opción al usuario de hacerla manualmente. Durante esta etapa del proyecto, el ingreso del número de matrícula se hará manualmente.

Necesidad: No negociable

Prioridad: Alta

Estabilidad: 0 de 0. No se esperan cambios a futuro.

Fuente: Entrevista del -mar-2003 con el Sr. Pérez

: El Sistema debe permitir el cálculo de los promedios, tanto trimestral como semestral (dependiendo de cada establecimiento). Para cada alumno debe existir el cálculo de los promedios semestrales (o trimestrales), promedio general (todas las notas hasta antes del examen), y por último, el promedio final (todas las notas incluyendo el examen).

Necesidad: No negociable

Prioridad: Alta

Estabilidad: 5 de 0. El cliente no se encuentra seguro. Se espera una especificación más detallada de la especificación del número de pruebas durante el período.

Fuente: Entrevista del 2 -mar-2003 con el Sr. uárez.

: El Sistema debe proveer mensajes de ayuda sobre cada función. El usuario decidirá si ocupa o no esta opción, de modo de no sobrecargarlo con demasiada información.

Necesidad: No negociable

Prioridad: Media

Estabilidad: 7 de 0. Se espera el DRS para definir adecuadamente cual funcionalidad deberá tener la ayuda.

Fuente: Entrevista del 5-may-2003 con el Sr. Ariza.

5.3.3 Problemas de los requerimientos de usuarios

El proceso de especificación de los requerimientos de usuario es iterativo, y termina cuando éstos se han estabilizado. No obstante, los requerimientos pueden tener algunos problemas clásicos. stos son:

- *orrectitu* : Es necesario que todos los requerimientos de usuario sean especificado en forma correcta con respecto a lo que el usuario desea, para poder cumplir con el objetivo que se pide.
- *ompletitu* : Es necesario asegurarse de que todos los requerimientos que se especificaron permiten definir completamente el sistema a desarrollar, y no falta ninguno de ellos. Esto es extremadamente complejo de realizar.
- *onsistencia*: Se debe mantener consistencia entre todos los requerimientos especificados. En otras palabras, ningún requisito especificado debe contradecir a ningún otro requisito o conjunto de requerimientos.
- *epen encia e los requerimientos*: Es deseable que todos los requerimientos sean independientes unos de otros. En caso de que esto no se cumpla, resulta conveniente marcar las dependencias entre los requerimientos. Esto debido a que si se desea modificar uno, será necesario revisar los requerimientos con dependencias para evitar introducir defectos.
- *mbi e a e los requerimientos*: Es muy importante que los requerimientos estén especificados en forma exacta, clara y no ambigua. Esto requiere un proceso conciente de revisión de cada uno de los requerimientos, con el propósito de asegurar una sola interpretación.
- *uplici a e los requerimientos*: Cada requisito debe ser especificado solamente una vez. Se debe evitar especificar varias veces un mismo requisito, dado que constituiría una pérdida de tiempo, y un múltiple control sobre un mismo requisito que ya fue controlado. No obstante, en caso de requerirse debido a facilidad de entendimiento de los requerimientos, es

posible duplicarlo. En dicho caso, se hace necesario introducir un atributo que permita mantener los identificadores de los requerimientos duplicados. En caso de modificar alguno de estos requerimientos, se modifican también sus copias.

5.3. Definición de Requerimientos de Soft are

Esta fase corresponde a la segunda fase del proceso de desarrollo, en donde también se realizan tareas de análisis. La fase de requerimientos de usuario corresponde a un proceso que se realiza desde el lado del usuario, utilizando su lenguaje y formas de especificar el problema a resolver. En dicha etapa, no se utilizó lenguaje técnico para describir el sistema a desarrollar. Ello, debido a que el usuario generalmente no tiene capacitación en el área informática. sto indica que es necesario tomar el documento DRU producido en la fase anterior, y traducirlo a especificaciones del sistema de software, utilizando lenguaje técnico. Por ello, a partir de los requerimientos de usuario, se debe construir un conjunto de requerimientos de software lo más completo, consistente y correcto posible. Esto es crucial, debido a que se están produciendo requerimientos que afectan directamente al software a construir.

Esta segunda fase contempla principalmente el desarrollo de los requerimientos del sistema, pero no desde el punto de vista del usuario, sino que del desarrollador. Dichos nuevos requerimientos se denominan requerimientos de software. Lo anterior indica que esta fase toma como entrada, el DRU, y produce un documento donde se expresan los requerimientos de software. Dicho documento se denomina "Documento de Requerimientos de Software", conocido también por su sigla DRS.

Es necesario tener una buena definición de requerimientos de software, porque éstos influirán directamente sobre la pieza de software entregada. La tarea realizada en esta fase es la de tomar cada uno de los requerimientos de usuario, traduciéndolos a requerimientos de software. Es posible (y recomendable) usar prototipos para esta fase de desarrollo del proyecto para poder ilustrar el avance que se lleva en el proyecto, asociando los requerimientos en cuestión. Por otra parte, también es positivo usar prototipos para medir el impacto que tiene cierto requisito sobre el proyecto.

Al igual que en la fase de definición de requerimientos de usuario, una de las salidas que tiene esta fase es el DRS. Este documento define los requerimientos de software descritos anteriormente. La fase finaliza al ser aprobado el DRS, actividad que se realiza después de la actividad de revisión de requerimientos de software (RS R). El proceso de aprobación del DRU es una actividad formal, que se realiza de acuerdo a un procedimiento establecido antes del inicio del proyecto.

5.3.5 Construcción del modelo conce tual

El modelo conceptual de un sistema corresponde a una definición jerárquica de las funciones de dicho sistema. Dicha especificación funcional debe ser independiente de la implementación del sistema.

Es posible construir el modelo conceptual por descomposición "top-down" de la función principal, lo cual debe ser inferido del DRU en una jerarquía de funciones. Para algunos tipos de proyecto, estas funcionalidades podrían por ejemplo venir dadas por distintos perfiles de usuarios asociados al proyecto, o descomposición en módulos, etc. Lo importante es que sea posible hacer una descomposición de la función principal en sub-funciones, de forma a poder atacar cada una de forma independiente. Nótese que ya a este nivel se puede notar la importancia de que los requerimientos de usuario conserven su independencia.

El proceso de modelación es un proceso que debe ser iterativo, pues requiere mucho desarrollo y cambios. El desarrollo de la especificación de las funciones se realiza por niveles, partiendo por la función principal hacia abajo.

El modelo conceptual debe satisfacer las siguientes reglas:

• *Las funciones deben tener un propósito único*. Sus nombres deben tener una estructura declarativa (qué hacen en vez de cómo lo hacen), lo cual hace que esta estructura se mantenga a nivel de modelo, y no corresponda a una implementación de la solución. Esto se hace para mantener un esquema de alto nivel del desarrollo del proyecto.

• *Las funciones deben estar en el nivel apropiado en el que aparecen*. Esto se refiere a que se debe mantener cierto orden cronológico en la implementación de la solución, y también un orden de prioridades con respecto al orden "topdown" definido anteriormente. Esto se hace para mantener consistencia en la solución del problema.

• *Las interfaces con otros sistemas de software deben minimizarse*. En esta etapa de desarrollo del proyecto, no interesa mayormente desarrollar dichas interfaces, sino que tratar la solución de fondo del problema, concentrándose en la descomposición de éste.

• *Se recomienda que cada función no se descomponga en más de siete subfunciones*. Esto para evitar complejidades en el manejo de dichas funciones.

• *El modelo debe omitir información de implementación*, porque solamente se está proponiendo una descomposición del problema, y no se está presentando la solución final.

• *Debe especificarse los atributos de rendimiento de cada función* (capacidad, velocidad de ejecución, tiempos de respuesta, etc.), con el fin de tener una medida aproximada del costo de la solución en términos de velocidad, de la capacidad de procesamiento del sistema, como de los parámetros físicos que el entorno del sistema debe soportar.

• *Se debe identificar las funciones críticas*. Una función crítica es aquella que por su naturaleza, requiere un tratamiento especial. Algunos aspectos que pueden hacer crítica a una función son su complejidad de diseño o implementación, rendimiento, seguridad, o conectividad con otros sistemas.

• *Es conveniente utilizar herramientas CASE* para construir el modelo conceptual. Esto no sólo ayuda a la especificación del modelo conceptual, sino que también a su futura manutención.

5.3.6 Especificación de los requerimientos de software

El estándar ESA PSS-05-0 define claramente los distintos tipos de requerimientos de software a considerar, dividiéndolos en catorce grupos distintos. Todos los requerimientos de software tienen que ver esencialmente con la parte de operación, funcionamiento, seguridad y mantenimiento del software. En total, dichos requerimientos especifican el software. Estos requerimientos son derivados de los requerimientos que fueron estipulados por el usuario en la fase de definición de requerimientos de usuario.

A continuación se definen cada uno de los catorce grupos que componen los requerimientos de software:

• *Requerimientos funcionales*: Éstos especifican qué debe hacer el software, los cuales son derivados del modelo conceptual.

• *Requerimientos de rendimiento*: Son valores numéricos para variables medibles. Pueden ser incluidos en la especificación cuantitativa de cada función, o especificados en forma independiente. Los atributos de rendimiento deben ser presentados como rangos de valores: peor caso aceptable; valor nominal a ser usado en la planificación; mejor caso, para indicar potencial de crecimiento.

• *Requerimientos de interfaces*: Estos requerimientos especifican hardware, software o elementos de bases de datos con los que el sistema o sus componentes interactúan o se comunican. Deben ser clasificados en requerimientos de software, hardware y comunicaciones. Pueden utilizarse diagramas de bloques para ilustrar parte de este tipo de requerimientos.

• *Requerimientos operacionales*: Son los que especifican la forma en que correrá el sistema y como se comunicará con los operadores humanos. Incluyen todas las interfaces de usuario, interacción humano-computador, y requerimientos logísticos y organizacionales.

• *Requerimientos de recursos*: Requerimientos que especifican los límites superiores de los recursos físicos tales como capacidad de procesamiento, memoria principal, espacio en disco, etc.

• *Requerimientos de verificación*: Estos requerimientos especifican las restricciones de cómo se verificará el software. Pueden incluir requerimientos de simulación, emulación, tests vivos con entradas simuladas, tests vivos con entradas reales, e interfaces con ambientes de testeo.

• *Requerimientos de aceptación de tests*: Requerimientos que deben especificar las restricciones de cómo se validará el software.

• *Requerimientos de documentación*: Requerimientos que especifican documentación adicional a la requerida en el estándar.

• *Requerimientos de seguridad de la información*: Son requerimientos para asegurar el sistema y sus datos contra amenazas a la confidencialidad, integridad y disponibilidad.

• *Requerimientos de portabilidad*: Son aquellos que especifican la facilidad para utilizar el sistema en otros computadores y sistemas operativos.

• *Requerimientos de calidad*: Requerimientos que especifican atributos del software para asegurar que estará saludable para sus propósitos. Cuando sea posible, se debe definir atributos de calidad que sean claramente medibles.

• *Requerimientos de confiabilidad*: Éstos especifican los tiempos medios entre fallas o la tasa de ocurrencia de fallas que serían aceptables.

• *Requerimientos de mantenimiento*: Son los requerimientos que especifican cuan fácil es reparar fallas y adaptar el software a nuevos requerimientos. Debiera especificarse en forma cuantitativa, tal como el tiempo medio para reparar una falla.

• *Requerimientos de seguridad de la operación*: Son aquellos que especifican cualquier requisito para reducir la posibilidad de daño que puede producirse de una falla del software.

Los atributos asociados a los requerimientos de software son los mismos que los de los requerimientos de usuario, lo único que se cambia es la clave del identificador por “RS x.”.

5.3.7 Ejemplos de requerimientos de software

A continuación se muestran dos ejemplos de Requerimientos de Software, uno funcional y uno de desempeño:

RS 04: El Sistema debe permitir que las fechas de inicio y término de cada año escolar (días de vacaciones, feriados) puedan ser modificadas por el administrador de cada colegio, teniendo en consideración que estas fechas pueden diferir para cada curso.

Necesidad: No negociable

Prioridad: Alta

Estabilidad: 4 de 10.

Fuente: RU 15

RS 38: El Sistema debe soportar una cantidad simultánea de usuarios acorde con la demanda estimada por la Corporación. Cabe señalar que el Sistema será probado

inicialmente en 2 colegios. Luego pretende ser implementado a corto plazo en 30 colegios, para después extender dicha cifra a 100 establecimientos. Vale la pena destacar que el número de establecimientos irá incrementándose de acuerdo a la necesidad de la comuna.

Necesidad: No negociable

Prioridad: Alta

Estabilidad: 10 de 10.

Fuente: RU 23

5.3.8 Problemas de los requerimientos de software

Una vez que se han definido los atributos de estos requerimientos, es necesario analizar algunos de los problemas que se deben evitar en los requerimientos:

• *Completitud de los requerimientos de software*: Es necesario considerar dos aspectos importantes: Si es que se han considerado todos los requerimientos de usuario dentro de la construcción de los requerimientos de software, y si es que se ha especificado una actividad para cada conjunto posible de entradas. En la medida que se pueda responder afirmativamente a estas dudas planteadas, se habrá hecho todo lo posible porque los requerimientos de software estén completos. La completitud quedará demostrada mediante una matriz de trazabilidad entre requerimientos de usuario y requerimientos de software. Esta matriz es parte del DRS. A continuación se muestra un ejemplo de dicha matriz:

RS \ RU	1	2	3	4	5	6	7	8	9	10
1							X			
2							X			
3							X			
4										
5									X	
6									X	

Fig. 5.4 Matriz de trazabilidad de requerimientos.

La matriz muestra en las columnas a los RU, y en las filas, a los RS. El ejercicio requiere realizar, para cada RU, una cruz en el o los correspondientes RS que lo especifican. Del ejemplo, se puede observar que el requisito de usuario 7 es especificado por los requerimientos de software 1, 2 y 3. Al realizar la matriz, es recomendable poner los RU en las columnas y los RS en las filas, debido a un problema de tamaño: el número de RS es considerablemente mayor que el de RU.

• *Correctitud de los requerimientos de software*: El proceso de traducción desde los requerimientos de usuario a requerimientos de software puede introducir defectos. Por ello, se hace necesario trabajar sobre dicho proceso, con el propósito de obtener requerimientos de software correctamente traducidos. Nótese que es de suma importancia que los requerimientos de usuario definan adecuadamente el sistema que desea el usuario. Por ello, la corrección radica en que el proceso de traducción de los RU no haya introducido ningún tipo de error o inconsistencia en su o sus correspondientes RS.

• *Consistencia de los requerimientos de software*: La consistencia de los requerimientos de software se da si es que ningún requisito entra en conflicto con algún otro. Con el fin de evitar estos conflictos, se nombrarán algunas de estas inconsistencias:

1. Términos diferentes para nombrar el mismo objeto.

2. El mismo término para nombrar diferentes objetos.
3. Actividades incompatibles pasando al mismo tiempo.
4. Actividades pasando en orden erróneo.

• *Duplicación de los requerimientos de software*: En general, este problema debe ser evitado, pero a veces es necesario, para llevar a un mejor seguimiento del DRS.

5. E E E E E E E E

Éste (algunas veces denominado especificación de requerimientos del software) es la declaración oficial de qué es lo que requieren los desarrolladores del sistema. Incluye tanto los requerimientos del usuario para el sistema como una especificación detallada de los requerimientos del sistema. En algunos casos, los dos tipos de requerimientos se integran en una única descripción. En otros, los del usuario se definen en una introducción de la especificación de los del sistema. Si existe un gran número de requerimientos, los detalles de los requerimientos del sistema se pueden presentar como documentos separados.

El documento de requerimientos tiene un conjunto diverso de usuarios que va desde los administradores principales de la organización, quienes pagan por el sistema, hasta los ingenieros responsables del software. La figura 5.16, tomada de (otonya y Sommerville, 1), muestra los posibles usuarios del documento y cómo lo utilizan.

El documento de requerimientos de software debe satisfacer seis requisitos:

- Especificará únicamente el comportamiento externo del sistema.
- Especificará las restricciones de la implementación.
- Será fácil de cambiar.
- Servirá como herramienta de referencia para los mantenedores del sistema.
- Registrará las previsiones del ciclo de vida del sistema.
- Caracterizará las respuestas aceptables para los eventos no deseados.

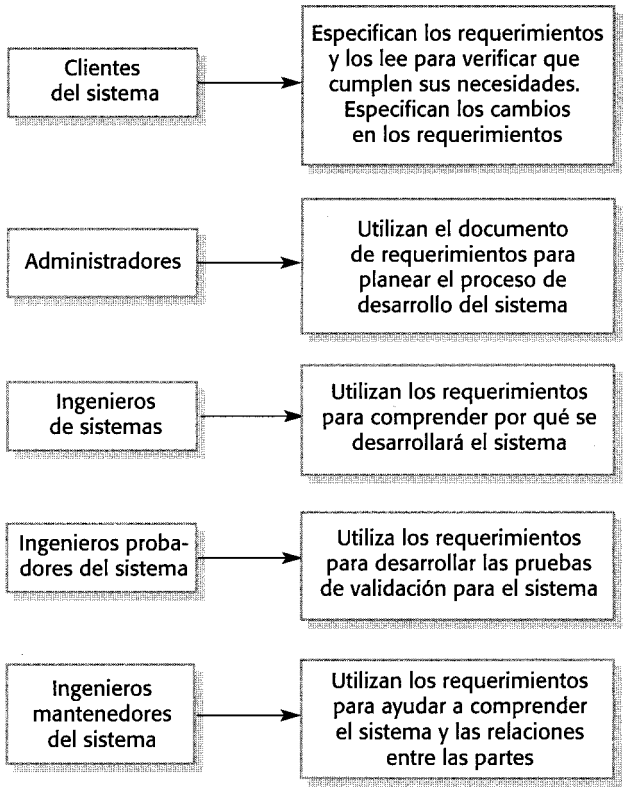
Clientes del sistema
Especifican los requerimientos y los lee para verificar que cumplen sus necesidades. Especifican los cambios en los requerimientos
Administradores
Utilizan el documento de requerimientos para planear el proceso de desarrollo del sistema
Ingenieros de sistemas
Utilizan los requerimientos para comprender por qué se desarrollará el sistema
Ingenieros proba-dores del sistema
Utiliza los requerimientos para desarrollar las pruebas de validación para el sistema
Ingenieros mantenedores del sistema
Utilizan los requerimientos para ayudar a comprender el sistema y las relaciones entre las partes

Figura 5.5 Usuarios de un documento de requerimientos

La ficina Nacional de Estándares de E.E.U.U (ANSI), el Departamento de Defensa de E.E.U.U, y la IEEE (estándar 30, 1 4) han propuesto formatos para las especificaciones de requisitos del Software. Algunos puntos propuestos son:

Introducción: Establece metas y objetivos del Software (Visión, isión del sistema, alcance, etc.).

Descripción de la nformación Proporciona una detallada descripción del problema que el software va a resolver. Se documenta el contenido de la información y sus relaciones, flujo y estructura. Se describen las interfaces ardware-Software-Usuarios (Interfaz ombre- áquina) del sistema.

Descripción uncional: Se describen todas las funciones requeridas para la solución del problema. Se proporciona una descripción del proceso de cada función; se establecen y justifican las restricciones del diseño; se establecen las características del rendimiento. Se especifica el comportamiento del sistema, es decir, como actuará el sistema ante acontecimientos externos y como son los mecanismos de control internos. Se anexarán diagramas de flujo de datos(DFD S), de flujo de control(DFC) y de transición de estados (DET) para describir gráficamente la Estructura lobal del Software y las interacciones entre las funciones del software y otros elementos del sistema.

Criterios de alidación. El tipo de pruebas que se efectuarán para validar una función, el rendimiento y las restricciones.

ibiografía. Referencias a los documentos de la Empresa o Institución que se consultaron para establecer los requisitos; a libros y artículos sobre el tema; y a estándares utilizados.

Apéndices. Tablas de Datos, Descripciones detalladas de Algoritmos, Gráficas, Dibujos, Prototipos de Manual de Usuario, etc.

ESQUEMA PARA LA ESPECIFICACIÓN DE REQUISITOS
(IEEE/ANSI 830, 1984)

I. INTRODUCCIÓN

A. Referencia del Sistema

B. Descripción general.

C. Restricciones del Proyecto de software (alcances del sistema).

II. DESCRIPCIÓN DE LA INFORMACIÓN.

A. Representación del contenido de la información.

B. Representación del flujo de la información.

1. Flujo de datos.
2. Flujo de control.

III. DESCRIPCIÓN FUNCIONAL.

A. Participación funcional, división en funciones principales y secundarias.

B. Descripción funcional.

1. Descripción del procesamiento.
2. Restricciones / limitaciones.
3. Requisitos de rendimiento.
4. Restricciones de diseño.
5. Diagramas de Soporte.

C. Descripción del control.

1. Especificación del control.
2. Restricciones de diseño.

DESCRIPCIÓN DEL COMPORTAMIENTO.

A. Estados del sistema.
B. Eventos y acciones.

CRITERIOS DE VALIDACIÓN

A. Límites del rendimiento.
B. Clases de pruebas.
C. Respuesta esperada del software.
D. Consideraciones especiales.

BIBLIOGRAFÍA.

VII. APÉNDICES.

ESQUEMA PARA LA ESPECIFICACIÓN DE REQUISITOS

(IEEE/ANSI 830, 1998)

1. Introducción
1.1. Propósito del documento de requerimientos
1.2. Ámbito o alcance del sistema
1.3. Definiciones, acrónimos y abreviaturas
1.4. Referencias
1.5. Visión general o resumen del documento

2. Descripción general
2.1. Perspectiva del producto.
2.2. Funciones del sistema
2.3. Características de los usuarios
2.4. Restricciones
2.5. Suposiciones y dependencias

3. Requisitos específicos
3.1. Requisitos funcionales
3.2. Requisitos de interfaces externos
3.2.1. Interfaces de usuario
3.2.2. Interfaces hardware
3.2.3. Interfaces software
3.2.4. Interfaces de comunicación
3.3. Requisitos de rendimiento
3.4. Requisitos de desarrollo
3.5. Requisitos tecnológicos
3.6. Atributos
3.7. Otros requisitos

4. Apéndices

Por ultimo presentamos otra propuesta de es uema, tomada de la página eb Plantilla de la specificaci n de e uisitos de oft are cu a direcci n es ttp polaris.lcc.uma.es amg P re uisitos. tml.

ESPECIFICACIÓN DE REQUISITOS DEL SOFTWARE

1.0 Introducción

n esta secci n se ace un resumen del contenido del documento.

1.1 Objetivos

Describir los objeti os generales del soft are.

1.2 Alcance

Describir el soft are en t rminos de entradas salidas principales funcionalidades principales sin entrar en detalles de implementaci n.

1.3 Contexto

l soft are se encuadra en una l nea de productos se discuten los aspectos estrat gicos de esa l nea. l objeti o es situar al lector acerle comprender el sistema como un todo.

1.4 Restricciones principales

e enumeran las restricciones de negocio ue puedan afectar a la especificaci n, dise o implementaci n o prueba del soft are.

2.0 Escenarios de uso

sta secci n organi a la informaci n de re uisitos ue se recoge del cliente.

2.1 Perfiles de usuario

Describir los usuarios en el sentido de actores del soft are.

2.2 Casos de uso

e presentan todos los casos de uso del sistema.

2.3 Consideraciones especiales

e describen si las a las consideraciones re uisitos relacionados con el uso del soft are.

3.0 Modelo de datos

sta secci n describe el dominio de informaci n del soft are.

3.1 Modelo de objetos

e describe el modelo de objetos esencial el deri ado directamente del dominio del problema. e describen tanto objetos como relaciones.

4.0 Modelo Funcional

sta secci n describe cada una de las funcionalidades principales del soft are.

4.1 Descripción de funciones

e detalla la descripci n de una de las funcionalidades principales. sta secci n se repite para cada funci n principal.

4.1.1 Especificación de proceso

e describe la funcionalidad te tualmente mediante diagramas de acti idad, colaboraciones, etc.

4.1.2 Rendimiento

e describen los re uisitos de rendimiento de la funcionalidad.

4.1.3 Restricciones de diseño

e describen las restricciones de dise o ue puedan afectar a la especificaci n, dise o implementaci n o prueba del soft are.

4.2 Descripción Interfaces

sta secci n detalla los interfaces del soft are con el mundo e terior .

4.2.1 Interfaces HW

e describen las interfaces con otras má uinas ordenadores o dispositi os .

4.2.2 Interfaces SW

e describen las interfaces con otros sistemas o productos .

4.2.3 Interfaces Humanas

e presenta una bre e descripci n de los interfaces con los usuarios umanos del soft are.

5.0 Modelo de comportamiento

sta secci n presenta los elementos principales del modelo de comportamiento.

5.1 Comportamiento software

e describen los e entos estados principales.

5.1.1 Eventos

ista de e entos principales.

5.1.2 Estados

ista de estados principales.

5.2 Diagramas de transición de estados

lo los del sistema soft are completo.

6.0 Limitaciones, y restricciones

e describen restricciones espec ficas ue puedan afectar a la especificaci n, dise o implementaci n o prueba del soft are.

7.0 Criterios de validación

nfo ue general ue se usará para la alidaci n del soft are, inclu endo tipos de pruebas, respuestas esperadas l mites de rendimiento admisibles.

8.0 Apéndices

Cualquier información suplementaria que complemente la especificación anterior.